DOCUMENT RESUME

ED 027 218 SE 006 293

Some Specifications for an Undergraduate Course in Digital Subsystems. An Interim Report of the Cosine Committee.

Commission on Engineering Education, Washington, D.C.
Spons Agency-National Science Foundation, Washington, D.C.
Pub Date Nov 68
Grant-NSF GY-2108
Note-11p.
EDRS Price MF-$0.25 HC-$0.65
Descriptors-*College Science, *Computers, *Course Descriptions, *Curriculum Development, Engineering, *Engineering Education
Identifiers-National Academy of Engineering, National Science Foundation

This report describes an undergraduate course in digital subsystems. The course is divided into two major parts. Part I is entitled Electronic Circuits and Functional Units. The material in this part of the course proceeds from simple understandings of circuits to the progressively more complex functional units. Early emphasis is placed on basic properties underlying digital circuits such as simple inverters and gates. The interconnection of these gates to perform simple logic functions is illustrated by detailed consideration of useful typical logic modules. The introduction of flip-flops then leads to sequential circuits such as registers, counters, and A-D and D-A converters. Interconnection of these sequential circuits leads to consideration of larger functional units. With the student now understanding the operations and properties of the basic functional units, he is ready to proceed to Part II, Digital Subsystem Design. Part II of the course is subdivided into two sections: (1) Design Principles and Procedures, and (2) Examples of Digital Subsystem Design. Section One contains a detailed outline of the many considerations in digital subsystem design and concluded by outlining a general design procedure. Section Two contains a graded set of examples of digital subsystems. A reference section is provided for each part of the course. (BC)

ED0 27218

Some Specifications For An Undergraduate Course In Digital Subsystems

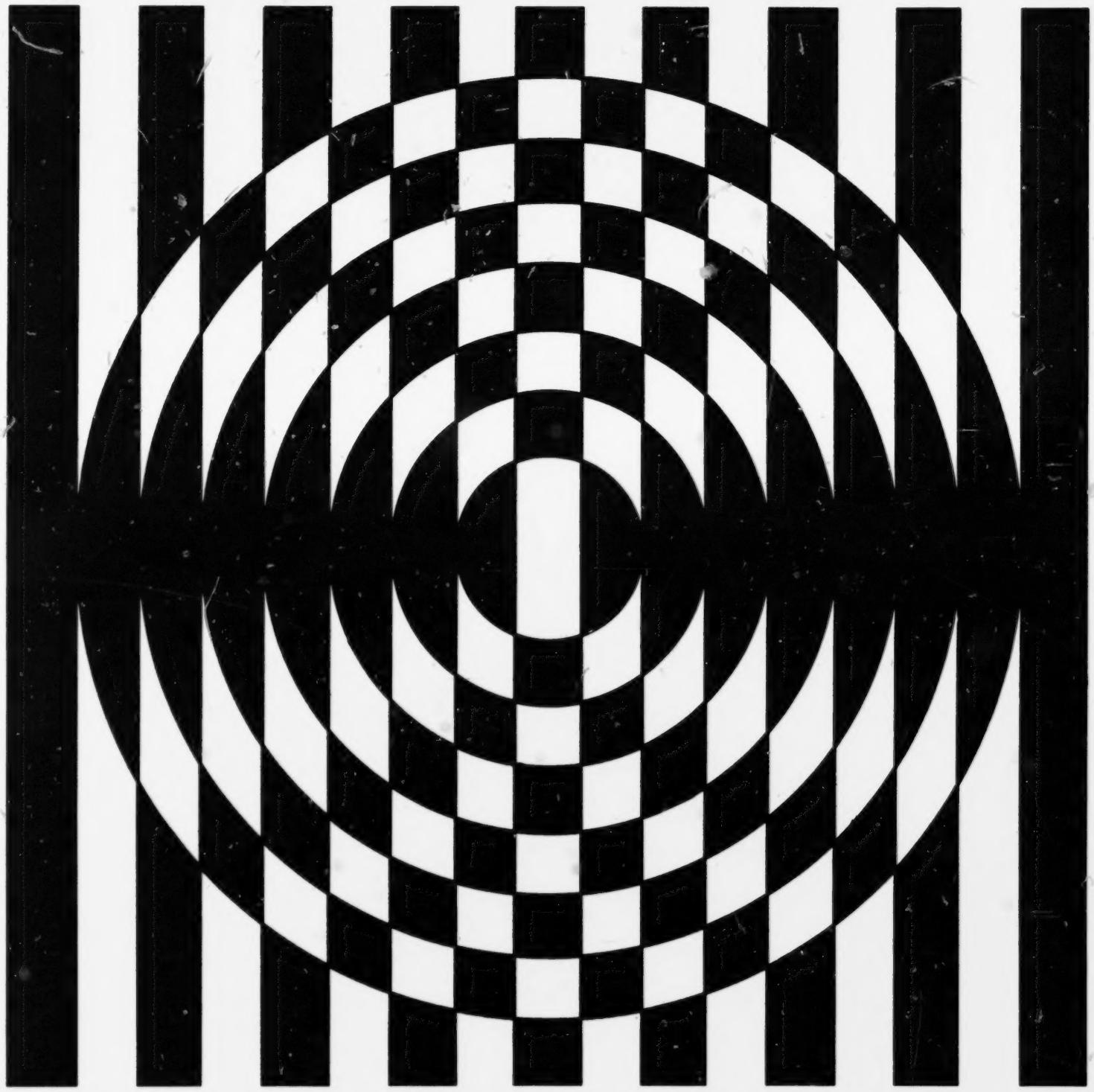
November 1968

U.S. DEPARTMENT OF HEALTH, EDUCATION & WELFARE
OFFICE OF EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL OFFICE OF EDUCATION POSITION OR POLICY.

Cosine Committee

Commission on Engineering Education



SE 006 293

SOME SPECIFICATIONS FOR AN UNDERGRADUATE COURSE IN DIGITAL SUBSYSTEMS

An Interim Report of the

COSINE COMMITTEE

of the

COMMISSION ON ENGINEERING EDUCATION

of the

NATIONAL ACADEMY OF ENGINEERING
2101 Constitution Avenue
Washington, D.C. 20418

November, 1968

This report has been supported in part by the National Science Foundation under Grant GY-2108.

TABLE OF CONTENTS

I. INTRODUCTION 5

II. GENERAL COURSE DESCRIPTION 5

III. PART I, ELECTRONIC CIRCUITS AND FUNCTIONAL UNITS 5

Preface 5

Outline of Course Material 5

References 7

IV. PART II, DIGITAL SUBSYSTEM DESIGN 7

Preface 7

Design Principles and Procedures 7

Examples of Digital Subsystem Design 9

More Extensive Examples 10

References 10

I. INTRODUCTION

Digital subsystems are well known as elements of general purpose digital computers. They also occur in many other special applications, such as communications, control, and laboratory instrumentation.

Recently, progress in electronics has made it possible to develop very complex systems by combining many different subsystems. The latter may perform such varied functions as computation, data formatting, buffering, synchronizing, switching, and digital filtering. Many of the special purpose digital subsystems will be custom designed. Taken together, they represent a major factor in system cost and a significant area of engineering effort.

The present report is offered to help widen the opportunities for Electrical Engineering undergraduates to study this important subject. The Digital Subsystems course is intended to be an elective course in Electrical Engineering at the Junior or Senior level.

Specifications for the course were developed by Task Force III of the COSINE Committee at and following a meeting during August 1-3, 1968 at Tarrytown, New York. The work has been supported by a grant from the National Science Foundation.

II. GENERAL COURSE DESCRIPTION

The course divides more or less sequentially into the following two parts:

(1) Electronic circuits and functional units
(2) Digital Subsystem Design
  (a) Design Principles and Procedures
  (b) Examples of Digital Subsystems Design

The task force was divided into two groups. One of these prepared the outline for Part 1, above, while the other worked on Part 2. Because of the natural sequence of the material, Part 1 can stand alone. Therefore, the reports of the two groups, which follow, are separately identified. However, Part 2 is best preceded by Part 1 in order to provide the student with a physical background and to help him develop an intuitive feeling for the relation between circuits and their functions.

It is hoped that all or most of the course material can be presented in one year. This will depend upon the depth of coverage and the number of hours. No attempt has yet been made to schedule the separate parts.

Recommended prerequisites are: a first course in Switching Theory (including Boolean algebra, minimization of combinational circuits, and an introduction to finite state machines) and a first course in electronics (including solid state components and simple circuits). In addition, it would be helpful for the student to have some acquaintance with computer programming, number systems, binary arithmetic, and communication systems.

No laboratory has been specifically designed for this course. It can readily be added by providing for the construction or simulation of some of the subject functional units and digital subsystems.

Since a general purpose undergraduate course in digital subsystems must satisfy the needs of a wide variety of students, such a course must cover a broad range of material, from basic digital circuits to complete digital subsystems. The problem facing this Task Force has been to select a course sequence which presents topics representative of the field, and to arrange these topics in a logical manner. It is not feasible to encompass the entire digital subsystems area, so care has been taken to select examples for study which illustrate methods used in subsystem design, rather than to catalog a large number of subsystem examples.

In spite of the large amount of material covered in the proposed course outline, it is not intended that this course be treated as a survey of the field. Instead, it is hoped that the student will gain enough insight into the design processes for digital subsystems that (with the help of some meaningful laboratory experience) he will be able to contribute significantly to their design.

Some of the material alluded to in the course outline may be either unfamiliar or unclear to potential instructors, so the committee has annotated the topic listings with references to commonly available books and articles. It is not intended that these references serve to delimit specific material to be covered, but rather that the references illustrate the intent of the individual outline entries.

III. PART 1
ELECTRONIC CIRCUITS AND FUNCTIONAL UNITS

Contributors:

C. L. Coates, University of Texas
W. G. Howard, Jr., University of California, Berkeley
A. W. Lo, Princeton University
G. A. Maley, IBM
E. J. McCluskey, Stanford University

Preface:

The material is organized to proceed from an understanding of the simplest circuits to the progressively more complex functional units. Early emphasis is placed on basic properties underlying digital circuits such as simple inverters and gates. The interconnection of these gates to perform simple logic functions is illustrated by detailed consideration of useful typical logic modules. The introduction of flip-flops then leads to sequential circuits such as registers, counters, and A-D and D-A converters. Interconnection of these sequential circuits then directly leads into consideration of larger functional units. The group felt that there were several important topics (i.e., memory arrays, packaging and interconnection problems) which do not directly fit into the hierarchical development. These are treated separately at the conclusion of Part 1, since they relate most closely to large arrays of circuits.

OUTLINE OF COURSE MATERIAL

I. *Basic Properties of Logic Elements* – The principles guiding the design of logic gate circuits and the basic terminal properties for specifying logic circuit performance. (Lo, Ch. 1)

A. Quantization – Input-Output Transfer Characteristics (Quantization Characteristics) (LMH,Ch. 6)

1. Requirements
  a. gain and threshold
2. Considerations
  a. tolerance
  b. noise
  c. fan-in, fan-out loading

B. Directivity
   1. Unilateral switching elements, undirectional coupling elements

C. Operating Speed
   1. Delay Times
   2. Rise and Fall Times
   3. Repetition Rate

II. Logic Gates – The electronic properties of the inverter (NOT) circuit and use of this circuit in the realization of elementary logic gates are described. The two dominant forms of implementation (DTL/TTL and ECL) are treated in detail.

A. Types of Gates (Lo, 2.3, 2.4)
   1. NOT (inverter, transistor inverter, electronic properties)
   2. NAND
   3. NOR
   4. OR
   5. AND
   6. Exclusive – OR

B. Implementation – DTL/TTL, ECL (LMH, Ch. 6)
   "Field Effect Transistors"

III. Combinational Subfunction Units – Functional description of commonly used logic modules and their realization by logic gates.

A. Single Output Units
   1. Three-Input Majority Circuit – A single output circuit with three inputs. The output is one when a majority (any 2 or all 3) of the inputs are 1. Functional expression is f(a,b,c) = ab +ac +bc. (Chu, 3-16)
   2. Parity Circuit – An n-input, one-output circuit realized with Exclusive - Or circuits. The function realized is: $f(x_1, x_2, \quad x_n) = x_1 \oplus x_2 \oplus \ldots x_n$

B. Multi-Output Units
   1. Half and Full Adder (M/E, pp.162-163) (G, 6.3.1)
   2. Encoder, decoder (matrices) (G, 4.1.4)
      (Chu, 9.2, 9.3,9.4,9.5)

IV. Sequential Subfunction Units – Basic logic units involving storage and their use in the realization of register type subfunction units.

A. Flip-Flops (G, 5.1.1) (Ph, pp.121-132)
   1. S-R Flip Flop (M/E, p. 183)
   2. T Flip-Flop (M/E, p. 266)
   3. J-K Flip-Flop – A two input (J-K) sequential network similar to the T flip-flop. When the two inputs are raised and lowered together the network, including the output lines, performs exactly as a T flip-flop. However, when the two inputs are raised independently, the network performs as a S-R flip-flop. The "J" input line takes on the characteristics of the "Set" line while the "K" input appears to be a "Reset" line.
   4. D Flip-Flop (Gated Latch) (M/E, p. 259)
   5. One-shot (M/E, p. 272)

B. Registers
   1. Register-to-Register data transfer (G, 6.1.1)(Chu, 10.3)
   2. Bussing (Hel. 5.1,5.2)
   3. Shift Registers (G, 6.1.3) (M/E, p. 271)

C. Counters
   1. Ring
   2. Binary (G, 6.2.2) (Chu, 16.4) (M/E, pp. 270-278)
   3. BCD (G, 6.2.2)

D. Single-Pulse Generator (synchronizer) (M/E, p. 261)

E. A-D and D-A Converters (L, pp. 739-747)

V. Function Units

A. Binary Parallel (G, 6.3.1)
   1. Addition (M/E, p. 160)
   2. Subtraction (M/E, p. 161) (G, 6.3.3)

B. BCD Parallel
   1. Addition (G, 6.3.2.1)

C. Codes and their conversion (Hel, 7.16-7.20)
   1. Parity Checking
   2. Error Correcting (R, pp. 185-190)
   3. Gray (G, 2.4.2)
   4. Alphanumeric

VI. Primitive Subsystems

A. Serial Arithmetic Units (R, pp.128-134)(G, pp. 6.3.1.1) (Chu, 12-2)

B. Binary to Decimal Converters (L, pp. 67-69)

C. Linear Sequential Circuits (Chu, pp. 313-314)

VII. Memories (Lo, 6.2,6.3)

A. Types of Memories
   1. Serial Memories
      a. Delay line (G, 8.3.3)
      b. Disk, or Drum (G, 8.3.4) (L, pp. 562-565)
      c. Tapes (Hel, 3.2)
   2. Random Access (Hel, 5.6)
      a. 2D (G, 8.3.1)
      b. 3D (G, 8.3.2)
      c. 2½D
   3. Associative Memories (G, 8.3.5)
   4. Read-Only Memories (Lo, pp. 205) (G, pp. 489-494)

B. Memory Technology
   1. Storage – Important Parameters:
      a. Holding Power
      b. Addressing
      c. Sensing
   2. Implementation
      a. Magnetic core and thin film memories (Lo, 6.4)
      b. Semiconductor memories – bipolar, IGFET (D. A. Hodges, "Large Capacity Semiconductor Memory," *Proc. IEEE* Vol. 56, No. 7 July, 1968 pp. 1148-1162)

VIII. Packaging

A. Speed

B. Interconnection Problem

C. Cost

D. Power Consumption

E. Part Number Problem (Inventory)

F. Integrated Circuit and Large-Scale Integration (R. N. Noyce, "Making Integrated Electronics Technology Work, *IEEE Spectrum*, Vol. 5, No. 5, May 1968, pp.63-66).

(J. A. Narud, C. D. Phillips, W. C. Seelbach, "Complex Monolithic Arrays: Some Aspects of Design and Fabrication" *Proc. NEC* Vol. 23, 1967 pp 303-308)

REFERENCES FOR PART 1

1. T. C. Bartee, *Digital Computer Fundamentals,* 2nd Edition McGraw-Hill Book Company, New York, 1966.

2. T. C. Bartee, I. L. Lebow, I. S. Reed, *Theory and Design of Digital Machines,* McGraw-Hill Book Company, New York, 1962.

3. Y. Chu, *Digital Computer Design Fundamentals,* McGraw-Hill, New York, 1962 (Chu).

4. H. W. Gschwind, *Design of Digital Computers,* Springer-Verlag, New York, 1967 (G).

5. H. Hellerman, *Digital Computer System Principles,* McGraw-Hill Book Company, New York, 1967, (Hel).

6. R. S. Ledley, *Digital Computer and Control Engineering* McGraw-Hill Book Company, New York, 1960, (L).

7. A. W. Lo, *Digital Electronics* Addision-Wesley, Reading, Mass. 1967 (Lo)

8. Lynn, Meyer, Hamilton, *Analysis and Design of Integrated Circuits* McGraw Hill Book Co., New York 1967 (LMH).

9. G. A. Maley, J. Earle, *Logic Design of Transistor* Computers Prentice-Hall, Englewood Cliffs, N.J., 1963, (M/E)

10. G. A. Maley, M. F. Heilweil, *Introduction To Digital Computers,* Prentice-Hall, Englewood Cliffs, N.J., 1968.

11. M. Phister, *Introduction to the Design of Digital Computers,* John Wiley & Sons, Inc., New York, 1963 (Ph).

12. R. K. Richards, *Arithmetic Operations in Digital Computers,* Van Nostrand, New York, 1955 (R).

13. A. K. Susskind, *Notes on Analog – Digital Conversion Techniques,* Technology Press, Cambridge, Mass. 1957.

IV. PART 2
DESIGN PRINCIPLES, PROCEDURES AND EXAMPLES

Contributors:

G. D. Bergland, Bell Telephone Laboratories, Inc.
J. F. Kaiser, Bell Telephone Laboratories, Inc.
M. Karnaugh, IBM
F. F. Lee, Massachusetts Institute of Technolgoy
D. E. Troxel, Massachusetts Institute of Technology

Preface:

With the student now understanding the operation and properties of the basic functional units he is ready to utilize these elements in an actual digital subsystem design. In order to do this he must clearly understand the design process and all its related considerations.

It is realized that it is not only difficult to formalize the creative activity of design but it is questionable just how far to go in doing so. The fact that there has been very little standardization thus far in the field of design aids and procedures points up the difficulties of the problem. For example, the logic of a particular subsystem design may be describable in terms of a concise language, such as a register transfer language.[18] This same language can then be used as a means to run simulations or "logic exercises" of the subsystem on a general purpose digital computer. Such a language would seem to offer the advantages of conciseness, precision, and the facilitation of establishing automated test procedures. However, it remains to be seen whether any one particular language will be generally accepted for this purpose.

In view of these problems one cannot regard the section on Design Principles and Procedures as completely self-sufficient but only illustrative of the type of approach required. This section begins with a detailed outline of the many considerations in digital subsystem design and concludes with the outlining of a general design procedure. This is followed by a section containing a graded set of examples of digital subsystems. It is intended to provide a reasonable context in which the teacher and the student can test their design procedures. These examples are also selected in order to give the student a reasonable appreciation of the variety and the power of special purpose digital subsystems. As remarked earlier, a laboratory course can also be based on these subsystems.

Design Principles and Procedures

The student must have a clear idea of the many considerations involved in the different steps of the design process. Some of these considerations are categorized below.

1. Environmental specifications:

A digital subsystem should be characterized in terms of its external interactions before its detailed internal design is undertaken. These interactions may be with the environment, or they may be decided in the context of a larger system architecture. Not only the subsystem behavior as a data processor, but its terminal electrical properties and environment should be specified. Some of the important features are: logical interactions, data format, synchronization, asynchronous controls, balanced or unbalanced signals, signal levels and modulation, terminal impedances, permissible delays in response, modes of

failure, power supply, temperature, vibration, atmosphere, and connectors. Not all of these factors will be of significance in all stages of design.

2. External algorithm:

The logical interactions of the subsystem with its environment (the external algorithm) may be specified in a variety of ways, including English text, a special problem oriented language (e.g., mathematics, Algol, Fortran, etc.), or by means of diagrams. Some examples of external algorithms should be given.

3. Internal organization:

A configuration of functional units (e.g., adders, shift registers, function generators) must be specified which can implement the desired external algorithm. This is commonly done by means of a block diagram. Other aids to description and evaluation include special languages and flow charts. A number of alternative configurations may be considered and compared in the light of cost of implementation, flexibility, and other system requirements. A good understanding of the available functional units and configurations is required.

4. Sequential machine design:

The selected configuration must be specified in more complete detail as a sequential machine. Typically, a description at this level will define all of the registers and external switches or binary inputs and it will define the sequential changes in internal registers under all allowable conditions. The means for such descriptions include state diagrams, flow charts, natural language, and special languages. Synchronization and asynchronous control signals should be verified as well as circumstances permit by study, simulation, or breadboard construction. Alternatives may be considered and evaluated.

5. Realization:

The combinational circuits of the desired sequential machine must be designed in terms of selected hardware. Other functional units, such as registers, waveform generators, etc., must also be selected. The characterization of the hardware, in terms of such properties as delay, rise times, constraints on fan-in and fan-out, power, temperature, etc., should be available. These characteristics, along with the designed configuration, should permit some performance analysis in terms of reliability estimates, worst case margins, and simulations. The system may need to be reconfigured at this stage in order to assure acceptable performance. Iterated debugging and evaluation lead eventually to design acceptance, prototype construction, and testing.

In the design of a digital system it is necessary to break the system down into smaller subsystems in such a manner that each subsystem performs a subset of functions and at the same time maintains a well defined interface with other subsystems. For very large systems, each subsystem may be continued to be broken down into a succession of subsystems until each becomes manageable in size and complexity. The last level must, of course, lead to a description in terms of the elementary building blocks so that the subsystem may be constructed, tested, and integrated into the system hierarchy.

It is intended that procedures like those described below shall be applied to one or more of the special digital subsystems to be treated in this course. This approach will make the design procedure much more meaningful to the student. Parallel laboratory work, if available, will also be helpful.

The following step-by-step description is addressed to the design of subsystems at the base level.

Step 1. Problem Definition

This involves a description, in English, of the interface relationship between the subsystem under consideration and the other entities. The description specifies desired performance characteristics and the various physical and economic constraints.

Since several different methods may be used in achieving the desired functional capability, Steps 2, 3, and 4 have to be carried out for each of the methods. Ingenuity and experience are needed in the creation of these methods and the discarding of obviously poor ones.

Step 2. Defining Methods of Solution

A written description of the algorithm to be used, expressed in either English, or flow-chart or some algorithmic language.

Step 3. Preparation of Block Diagram

A block diagram is prepared which will show the major registers, memory elements and the major transfer paths.

Step 4. Reasonableness TEST

This involves a rough test to determine whether the configuration created during Step 3 can meet the performance requirement with the basic building blocks at the disposal of the designer, and whether the cost is within the acceptable limit.

Step 5. Selection Among Alternatives

The scheme which is judged most desirable from a cost, performance and component availability point-of-view is selected for the more detailed realization in the remaining steps.

Step 6. Register Transfer Description

This step is an elaboration of Step 3. It may be done by either making the block diagram more detailed, showing the control lines and timing for the various transfer paths, or it may be an expression of the subsystem in terms of a register transfer language. All the signals which have to be referred to in the description will be named from this point on.

Step 7. Logic Checking

If a register transfer language were used in Step 6, and if a simulator is available for the language, logical checking can be performed readily with the simulator, otherwise manual checking would be necessary. When errors are discovered, Step 6 is re-entered.

Step 8. Detail Logic Design

From the description of Step 6, logical expressions can be generated for the register transfer gates and the control of counters and flip-flops. The exact logical module types must be known at this time and the Boolean simplification must be done with the fan-in, fan-out, speed and available logic functions in mind.

This completes the design aspect of the realization of the subsystem. Breadboarding, construction and testing are the next steps which may reflect into reperforming some or all of the design steps outlined. Consideration must be given to the test of the subsystem at the time of design. Additional hardware may have to be provided so that the subsystem can be checked out.

Examples of Digital Subsystem Design

As the ultimate goal of the design of digital subsystems is to produce useful, reliable hardware, we feel that some exposure to the end product is necessary. Any course concerned with the design of digital subsystems should include classroom demonstrations of physical realizations of subsystems such as those described below.* Perhaps initially only one such subsystem might be available for classroom use; additional demonstrations would be developed for subsequent offerings of the course. Ideally, students (individually or in small groups), should carry out the design of a subsystem as a project, including the construction (using available logic modules), necessary debugging, and interfacing with other systems.

1. Shift Register Sequence Generators[1,2]

Using only shift registers and exclusive OR circuits, binary sequences of 1's and 0's can be generated. Some applications of these sequences are encipherment, error correcting codes, prescribed sequence generators, and pseudo-random bit generators.

The logical design of these subsystems is relatively easy. A sequence generator can be accompanied by a matched filter which can, for example, operate on the output of the sequence generator and detect the occurrence of specific subsequences. This can be particularly useful in illustrating the difficulty of detecting erroneous operation of a digital subsystem by means of oscilloscope traces.

2. Encoders and Decoders[1,2]

An encoder operates on a message word (either serial or parallel form) and produces an output message word which usually has a greater number of bits than the input word. A decoder reverses this operation with the added potential of correcting errors which may have occurred in transmitting the encoder output.

Examples of encoders and decoders for the detection and for correction of errors form a next level of complexity. While it is possible to generate (for example see p. 18, Chap. 2 of Golomb) a code for single error correction with a shift register, the message data is not always conveniently available in serial form. The efficacy of error correcting systems can be demonstrated by interposing a noisy channel between encoder and decoder and using a sequence generator and matched filter to provide a message source and sink.

3. PCM Transmission Systems[4]

A PCM system allows the introduction of analog interfaces with digital circuits. Discussions of A-to-D conversion techniques[5] can include such topics as timing, synchronization, iterative arrays, counters, and serial or parallel outputs. Normally, in a PCM system, the data samples are transmitted serially necessitating some synchronization scheme. The complexity of the receiver is a strong function of the synchronization scheme used. There is ample room for further expansion by including some multiplexing technique for multichannel operation.

4. Waveform Generators and Transversal Filters

A waveform can be described as a sequence of numbers or sample values. It may be generated by loading these sample values in a shift register or other form of memory and then strobing the memory sequentially to output the sequence.

If now the simple arithmetic operations of addition and multiplication are added to the delay operation of the shift register or strobed memory, the elementary "transversal filter" weighting function operator or "nonrecursive digital filter" results.

Each output sample is a weighted linear combination of $N + 1$ samples of the input sequence; i.e. the output sequence is obtained by convolving the input sequence with the weighting function of the filter. The a_i coefficients may be fixed or variable with time. This subsystem finds wide use in many data processing situations such as data smoothing, interpolation, equalization, and signal property estimation. The arithmetic operations can be performed either in serial or parallel form and to a precision dependent on the problem specification.

A particularly interesting simplification of the above filter occurs when all the a are ± 1. This obviates the need for the multiplication operation thus simplifying the subsystem design. Cascades of these simple filters[6] can produce higher order filters as the composite weighting function is the convolution of the subsidiary weighting functions.

5. Digital Filters

By adding to the basic transversal filter one or more signal feedback paths a general digital filter subsystem results.[6,7,8] One very useful simplification is the fundamental second-order digital filter section. By time sharing or multiplexing[9] this basic second-order section, digital filters of higher order may be realized rather simply. These complex filters find application as signal processors, control system compensators, smoothing filters, band elimination filters, signal property estimators, etc. The multiplexing approach also makes possible much more efficient utilization of the digital elements (gates, flip flops) especially for low data rate input signals.

The design of a basic second-order digital filter section furnishes an excellent vehicle for the study of arithmetic realization, timing, simple multiplexing with its required control circuits, effects of finite arithmetic (product truncation and round-off errors), quantization of signals, filter signal-to-noise ratios and scaling.

6. Correlator

A subsystem to compute the cross correlation function of two signals or the autocorrelation function of a single signal can be centered around a serial memory (shift register, delay line, or magnetic drum) and an arithmetic unit to compute the sum of products required. An interesting variation can be introduced at this time by utilizing a delay line (e.g. magnetostrictive) for the serial memory. This involves data formatting and synchronization to effect the storage and retrieval of signal samples. A-to-D converters[5] are required to interface to the input signals. An arithmetic unit[1,2] is required to compute a sum of delayed products of either the signal with itself or another signal. The operations here are very similar to those in a transversal filter. The major difference is that the use of a

*These representative examples are listed in order of increasing degree of complexity.

serial memory allows a single multiplier to be used while requiring a longer processing time and more complex control.

More Extensive Examples

Some examples of a more elaborate nature that are very characteristic of current design interest and that represent logical extensions of the design process are the alphanumeric display terminal and the Fourier analyzers. The alphanumeric display terminal might be a possibility if the necessary auxiliary equipment is available. The Fourier analyzer represents a more complex project which would also introduce the student to interfacing problems. When operational the analyzer can provide a very useful analysis tool for the laboratory.

1. Alphanumeric Display Terminal

An alphanumeric display[10] on a CRT requires local character storage so that a flicker-free visual image can be maintained.

The serial memory used for the correlators above can, of course, be used for storing other types of data. Character codes can be stored and CRT displays realized with the addition of a character generator. The data source is now not an A-to-D converter but a keyboard (typewriter, teletype, etc.). A question now arises as to what position (in memory or on the CRT) should be occupied by successive input characters.

A reasonable solution is to introduce a cursor on the CRT to indicate this position and allocate specific character codes to control the cursor movement. The availability of a light pen,[10] would allow the introduction of a digital feedback control system so that a cursor might be moved by the light pen. Another level of complication might be to provide some basic editing capabilities in addition to the ability to insert characters. These might include deletion of character, word, or line, justification, search for character strings, print out, etc.

2. Fourier Analysis Subsystems

With the advent of various Fast Fourier Transform (FFT) algorithms[11,12,13], a number of people have proposed implementing digital, real-time spectrum analyzers[14,15,16].

The basic set of arithmetic operations performed throughout the original Cooley-Tukey algorithm[11], involves a complex addition, subtraction and multiplication. The operands are accessed in a very regular pattern[17] which means that the indexing can be implemented fairly conveniently. In building FFT hardware, however, it is also worth considering radix 2, 4, 8, etc., algorithms[13].

Two possible FFT subsystems which could be built, are outlined, below. Both of these represent relatively ambitious projects, but they do serve to illustrate a wide variety of different design problems.

A. A Stand Alone Fast Fourier Transform Processor

The design of this type of a Fast Fourier Transform subsystem would involve

1. The use of A/D and D/A converters
2. Synchronization of the input, computations, and displays
3. Design of a high performance arithmetic and indexing unit, and
4. Interconnection of several subsystems

The arithmetic unit would perform the high speed multiply and addition operations on complex operands accessed from memory; the results are then stored back in memory. The operation of the indexing and control unit involves forming the patterns of operand and table memory addresses required in sequencing through the algorithm. The type of memory, arithmetic unit, A/D converter, display, and control unit will of course be very dependent on the input data rate assumed.

B. A Fast Fourier Transform Attachment for a Small General Purpose Computer

A small signal processing facility which repeatedly makes use of Fourier analysis or synthesis, might find it advantageous from cost and speed considerations to connect a relatively small FFT (64 point or so) subsystem to its central computer. If the FFT subsystem performed a 64 point FFT, for example, then the computer could perform an 8192 point transform using two radix 64 iterations in the FFT processor and one radix 2 iteration in the computer. Since the radix 64 iterations would be performed very rapidly, the overall execution time should be reduced significantly.

This form of processing system involves an even wider range of problems. First the interconnection and control of the processor and computer will introduce interfacing problems in both the hardware and the software. Also, even though the FFT processor assumed is quite small, most of the functions discussed for the stand alone unit must still be performed. Finally, the allocation of control functions between the computer and the FFT processor will introduce the added complexity of a hardware-software tradeoff.

REFERENCES FOR PART 2

1. Burtee, T. C., Lebow, I. L., and Reed, I.S., *Theory and Design of Digital Machines*, McGraw-Hill, 1962.
2. Hellerman, H., *Digital Computer System Principles*, McGraw-Hill, 1967.
3. Maley, G. A. and Hellwell, W.F., *Introduction to Digital Computers*, Prentice-Hall, 1966.
4. Hartley, G. C. et al, *Techniques of Pulse-Code Modulation in Communication Networks*, Cambridge University Press, 1967.
5. Susskind, A., *Notes on Analog-Digital Conversion Techniques*, Wiley, 1957.
6. Blackman, R.B., *Linear Data-Smoothing and Prediction in Theory and Practice*, Addison-Wesley, 1965.
7. Kuo, F.F. and Kaiser, J.F., *System Analysis by Digital Computer*, Wiley, 1966. See Chapter 7 on Digital Filters.
8. Gold, B. and C. Rader, "Digital Filter Design Techniques in the Frequency Domain," *Proc. IEEE* Vol. 55, No. 2, Feb. 1967, pp. 149-171.
9. Jackson, L. B., Kaiser, J. F., and McDonald, H.S., An Approach to the Implementation of Digital Filters, *IEEE Trans. Audio and Electroacoustics*, Sept, 1968.
10. Miller, J. C. and Wine, C. M., A Simple Display for Characters and Graphics, *IEEE Transactions on Computers*, Vol. C-17, No. 5, May 1968, pp. 470-475.

Miller, J. C. and Wine, C. M., A Light Pen for Remote Time Shared Graphic Consoles. *IEEE Transactions on Computers,* Vol. C-17, No. 8, Aug. 1968, pp. 799-802.

Christensen, C. and Pinson, E.N., Multifunction Graphics for a Large Computer System. *Proc. Fall Joint Computer Conference,* 1967, Vol. 31, pp. 697-711.

11. Cooley, J. W., and Tukey, J. W., "An Algorithm for the Machine Calculation of Complex Fourier Series", *Mathematics of Computation,* Vol. 19, pp. 297-301, April, 1965.

12. Cooley, J. W., "Complex Finite Fourier Transform Subroutine", *Share Document 3465,* September 8, 1966.

13. Gentleman, W. M., and Sande, G., "Fast Fourier Transforms – For Fun and Profit," *Proceedings of the 1966 Fall Joint Computer Conference (AFIPS),* Washington, Spartan Books, 1966.

14. Bergland, G. D., and Hale, H.W., "Digital Real-Time Spectral Analysis", *IEEE Transactions on Electronic Computers,* Vol. EC-16, pp. 180-185, April 1967.

15. Freudberg, R., DeLellis, J., Howard, C., and Schaffer, H. "An All Digital Pitch Excited Vocoder Technique Using the FFT Algorithm". *1967 Conference on Speech Communication and Processing Preprints,* pp. 297-310, November, 1967.

16. Shively, R.R., "A Digital Processor to Generate Spectra in Real-Time." *IEEE Transactions on Electronic Computers,* Vol. EC-17.

17. G-AE Subcommittee on Measurement Concepts, "What is the Fast Fourier Transform?" *IEEE Transactions on Audio and Electroacoustics,* Vol. AU-15, pp. 45-55, June 1967. Reprinted in *Proceedings of the IEEE,* Vol. 55, pp. 1664-1674, October 1967.

18. Duley, J. R. and Dietmeyer, D. L., "A Digital System Design Language (DDL), "*IEEE Transactions on Computers,* Vol. C-17, No. 9, September 1968, pp. 850-861.